

Python: module vcs.scatter

vcs.scatter

[index](#)

Scatter (GSp) module

Modules

[vcs.Canvas](#)

[vcs. vcs](#)

[vcs.queries](#)

[vcs.VCS validation functions](#)

[cdtime](#)

Classes

[builtin .object](#)

[GSp](#)

class **GSp**([builtin .object](#))

Class: [GSp](#)

Scatter

Description of [GSp](#) Class:

The Scatter graphics method displays a scatter plot of two 4-dimensional arrays, e.g. A(x,y,z,t) and B(x,y,z,t). The example below shows how to change the marker attributes of a scatter plot.

This class is used to define an scatter table entry used in VCS, used to change some or all of the scatter attributes in an existing table entry.

Other Useful Functions:

a=vcs.init()	# Constructor
a.show('scatter')	# Show predefined scatter graphics
a.show('marker')	# Show predefined marker objects
a.setcolormap("AMIP")	# Change the VCS color map
a.scatter(s1, s2, s,'default')	# Plot data 's1' and 's2' with 's' and 'default' template
a.update()	# Updates the VCS Canvas at the end of the command
a.mode=1, or 0	# If 1, then automatic update, if 0, then use update function to update the VCS Canvas.

Example of Use:

a=vcs.init()

To Create a new instance of scatter use:

sr=a.createscatter('new','quick') # copies content of 'quick' to

```

sr=a.createscatter('new') # copies content of 'default' to 'new'

To Modify an existing scatter use:
sr=a.getscatter('AMIP_psl')

sr.list()                                # Will list all the scatter
sr.projection='linear'                   # Can only be 'linear'
lon30={-180:'180W',-150:'150W',0:'Eq'}
sr.xticlabels1=lon30
sr.xticlabels2=lon30
sr.xticlabels(lon30, lon30)             # Will set them both
sr.xmtics1=''
sr.xmtics2=''
sr.xmtics(lon30, lon30)                # Will set them both
sr.yticlabels1=lat10
sr.yticlabels2=lat10
sr.yticlabels(lat10, lat10)              # Will set them both
sr.ymtics1=''
sr.ymtics2=''
sr.ymtics(lat10, lat10)                 # Will set them both
sr.datawc_y1=-90.0
sr.datawc_y2=90.0
sr.datawc_x1=-180.0
sr.datawc_x2=180.0
sr.datawc(-90, 90, -180, 180)          # Will set them all
sr.xaxisconvert='linear'
sr.yaxisconvert='linear'
sr.xyscale('linear', 'area_wt')         # Will set them both

```

Specify the marker type:

```

sr.marker=1                               # Same as sr.marker='dot'
sr.marker=2                               # Same as sr.marker='plus'
sr.marker=3                               # Same as sr.marker='star'
sr.marker=4                               # Same as sr.marker='circle'
sr.marker=5                               # Same as sr.marker='cross'
sr.marker=6                               # Same as sr.marker='diamond'
sr.marker=7                               # Same as sr.marker='triangle'
sr.marker=8                               # Same as sr.marker='triangledown'
sr.marker=9                               # Same as sr.marker='triangleright'
sr.marker=10                              # Same as sr.marker='triangleleft'
sr.marker=11                              # Same as sr.marker='square'
sr.marker=12                              # Same as sr.marker='diamondleft'
sr.marker=13                              # Same as sr.marker='triangleleftdown'
sr.marker=14                              # Same as sr.marker='triangleleftup'
sr.marker=15                              # Same as sr.marker='trianglerightdown'
sr.marker=16                              # Same as sr.marker='trianglerightup'
sr.marker=17                              # Same as sr.marker='squareleft'
sr.marker=None                           # Draw no markers

```

There are four possibilities for setting the marker color index

```

sr.markercolors=22                         # Same as below
sr.markercolors=(22)                       # Same as below

```

```

sr.markercolors=([22])                      # Will set the markers to a
                                              # color index
sr.markercolors=None                         # Color index defaults to Bla

To set the Marker size:
sr.markersize=5
sr.markersize=55
sr.markersize=100
sr.markersize=300
sr.markersize=None

```

Methods defined here:

__init__(self, parent, GSp_name=None, GSp_name_src='default', createGSp=0)

datawc(self, dsp1=1e+20, dsp2=1e+20, dsp3=1e+20, dsp4=1e+20)

list(self)

```

#####
#
# List out text combined members (attributes).
#
#####

```

rename = renameGSp(self, old_name, new_name)

```

#####
#
# Function:      renameGSp
#
# Description of Function:
#      Private function that renames the name of an existing
#      graphics method.
#
#
# Example of Use:
#      renameGSp(old_name, new_name)
#              where: old_name is the current name of scatter
#                      new_name is the new name for the scatter
#
#####

```

script(self, script_filename=None, mode=None)

Function: script # Calls _vcs.s

Description of Function:

Saves out a scatter graphics method in Python or VCS to
designated file.

Example of Use:

script(scriptfile_name, mode)

where: scriptfile_name is the output name of the

mode is either "w" for replace or "a" for append.

Note: If the the filename has a ".py" at the end of the string, then a Python script will be produced. If the filename has a ".scr" extension, then a VCS script will be produced. If neither extension is present, then by default a Python script will be produced.

```
a=vcs.init()
scr=a.createboxfill('temp')
scr.script('filename.py')                      # Append to a Python file
scr.script('filename.scr')                     # Append to a VCS file
scr.script('filename', 'w')

xmtics(self, xmt1=", xmt2=")

xticlabels(self, xtl1=", xtl2=")

xyscale(self, xat=", yat=")

yntics(self, ymt1=", ymt2=")

yticlabels(self, ytl1=", ytl2=")
```

Properties defined here:

datawc_calendar

```
get">get = _getcalendar(self)
set">set = _setcalendar(self, value)
```

datawc_timeunits

```
get">get = _gettimeunits(self)
set">set = _settimeunits(self, value)
```

datawc_x1

```
get">get = _getdatawc_x1(self)
set">set = _setdatawc_x1(self, value)
```

datawc_x2

```
get">get = _getdatawc_x2(self)
set">set = _setdatawc_x2(self, value)
```

datawc_y1

```
get">get = _getdatawc_y1(self)
set">set = _setdatawc_y1(self, value)
```

datawc_y2

```
get">get = _getdatawc_y2(self)
set">set = _setdatawc_y2(self, value)
```

marker

```
get">get = _getmarker(self)
set">set = _setmarker(self, value)
```

markercolor
get">*get* = _getmarkercolor(self)
set">*set* = _setmarkercolor(self, value)

markersize
get">*get* = _getmarkersize(self)
set">*set* = _setmarkersize(self, value)

name
get">*get* = _getname(self)
set">*set* = _setname(self, value)

projection
get">*get* = _getprojection(self)
set">*set* = _setprojection(self, value)

xaxisconvert
get">*get* = _getxaxisconvert(self)
set">*set* = _setxaxisconvert(self, value)

xmtics1
get">*get* = _getxmtics1(self)
set">*set* = _setxmtics1(self, value)

xmtics2
get">*get* = _getxmtics2(self)
set">*set* = _setxmtics2(self, value)

xticlabels1
get">*get* = _getxticlabels1(self)
set">*set* = _setxticlabels1(self, value)

xticlabels2
get">*get* = _getxticlabels2(self)
set">*set* = _setxticlabels2(self, value)

yaxisconvert
get">*get* = _getyaxisconvert(self)
set">*set* = _setyaxisconvert(self, value)

ymtics1
get">*get* = _getymtics1(self)
set">*set* = _setymtics1(self, value)

ymtics2
get">*get* = _getymtics2(self)
set">*set* = _setymtics2(self, value)

yticlabels1
get">*get* = _getyticlabels1(self)
set">*set* = _setyticlabels1(self, value)

```
yticlabels2
    get">get = _getyticlabels2(self)
    set">set = _setyticlabels2(self, value)
```

Data and other attributes defined here:

```
__slots__ = ['setmember', 'parent', 'name', 'g_name', 'yaxisconvert', 'xaxisconvert', 'marker', 'markers',
'projection', 'xticlabels1', 'xticlabels2', 'yticlabels1', 'yticlabels2', 'xmtics1', 'xmtics2', 'ymtics1', 'ymtics2',
'datawc_x2', ...]
```

```
g_name = <member 'g_name' of 'GSp' objects>
```

```
parent = <member 'parent' of 'GSp' objects>
```

```
setmember = <member 'setmember' of 'GSp' objects>
```

Functions

```
getGSpmember(self, member)
#####
#
# Function:      getGSpmember
#
# Description of Function:
#     Private function that retrieves the scatter members from the
#     structure and passes it back to Python.
#
#
# Example of Use:
#     return_value =
#         getGSpmember(self, name)
#             where: self is the class (e.g., GSp)
#                     name is the name of the member that is being
#                     retrieved.
#
#####
#
getmember = getGSpmember(self, member)
#####
#
# Function:      getGSpmember
#
# Description of Function:
#     Private function that retrieves the scatter members from the
#     structure and passes it back to Python.
#
#
# Example of Use:
#     return_value =
#         getGSpmember(self, name)
```

```

#           where: self is the class (e.g., GSp)
#           name is the name of the member that is being
#
# ##### ##### ##### ##### ##### ##### ##### ##### #####
renameGSp(self, old_name, new_name)
##### ##### ##### ##### ##### ##### ##### ##### #####
#
#   Function:      renameGSp
#
# Description of Function:
#     Private function that renames the name of an existing scatter
#     graphics method.
#
#
# Example of Use:
#     renameGSp(old_name, new_name)
#           where: old_name is the current name of scatter graph
#                   new_name is the new name for the scatter graph
#
##### ##### ##### ##### ##### ##### ##### ##### #####
setGSpmember(self, member, value)
##### ##### ##### ##### ##### ##### ##### ##### #####
#
#   Function:      setGSpmember
#
# Description of Function:
#     Private function to update the VCS canvas plot. If the value
#     set to 0, then this function does nothing.
#
#
# Example of Use:
#     setGSpmember(self, name, value)
#           where: self is the class (e.g., GSp)
#                   name is the name of the member that is being
#                   value is the new value of the member (or att
#
##### ##### ##### ##### ##### ##### ##### ##### #####
setmember = setGSpmember(self, member, value)
##### ##### ##### ##### ##### ##### ##### ##### #####
#
#   Function:      setGSpmember
#
# Description of Function:
#     Private function to update the VCS canvas plot. If the value
#     set to 0, then this function does nothing.
#
#
# Example of Use:
#     setGSpmember(self, name, value)

```

```
#           where: self is the class (e.g., GSp)
#           name is the name of the member that is being
#           value is the new value of the member (or att
# #####
```

Data

```
StringTypes = (<type 'str'>, <type 'unicode'>)
```